4.2 Linear Classification

In this section we are dealing with **linear classification**. We will primarily deal with the binary classification case but make some comments on the multi-class case when it is necessary. In particular, one can always use the one-vs-all or one-vs-one schemes introduced in Section 2.1.3 in order to solve the multi-class problem by decomposing it into a set of binary problems.

In linear binary classification we linearly separate the two classes as can be seen in Figure 4.4. Let $\mathcal{X} = \mathbb{R}^d$ be the input space, then the classifier $f : \mathbb{R}^d \to \{-1, 1\}$ has the form

$$f(x) = \mathrm{sign}(\langle w, x\rangle + b) = \begin{cases} 1 & \text{if } \langle w, x\rangle + b > 0, \\ -1 & \text{if } \langle w, x\rangle + b \leq 0. \end{cases}$$

This implies that we separate the input space into two half-spaces, one which we classify as positive and one which we classify as negative. We will call a training set $T = (X_i, Y_i)_{i=1}^n$ **linearly**

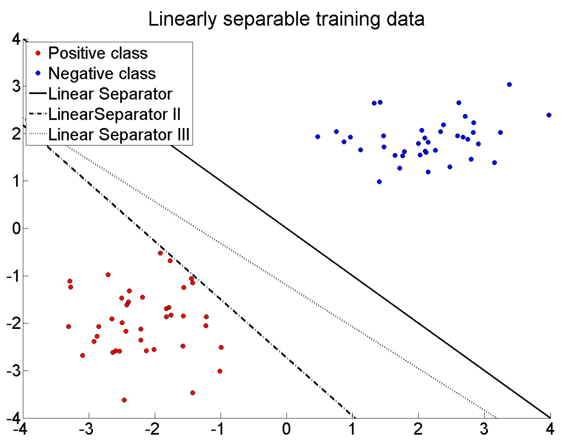


Figure 4.4: A training sample of a two-class problem in $\mathbb{R}^2$. The two classes are linearly separable and three different decision hyperplanes are shown which separate the two classes.

separable if there exists a weight vector w and an offset b such that,

$$Y_i\, f(X_i) = Y_i\, (\langle w, X_i\rangle + b) > 0, \qquad \forall i = 1, \ldots, n,$$

which simply means that there exists a hyperplane $\{x \in \mathbb{R}^d \mid \langle w, x\rangle + b = 0\}$ which separates the sets $X_+ = \{X_i \in T \mid Y_i = 1\}$ and $X_- = \{X_i \in T \mid Y_i = -1\}$.

We will make in this chapter no distinction between the original input space $\mathcal{X} = \mathbb{R}^d$ and a possibly larger **feature space**, where we first map the data via some transformations ϕ_i

$$x \in \mathbb{R}^d \longrightarrow (\phi_1(x), \ldots, \phi_D(x)),$$

to the feature space $\mathbb{R}^D$. As in the regression case we will always be linear in the new features $Z_i = \phi_i(x)$, $i = 1 \ldots, D$, which can result in a nonlinear classifier f in the original input space $\mathcal{X} = \mathbb{R}^d$, that is

$$f(x) = \mathrm{sign}(\langle w, \phi(x)\rangle + b) = \mathrm{sign}\Big(\sum_{j=1}^{D} w_j \phi_j(x) + b\Big).$$

Definition 14 *Let $g : \mathcal{X} \to \mathbb{R}$ be a function and $f(x) = \operatorname{sign}(g(x))$ be the resulting classifier with output in $\mathcal{Y} = \{-1, 1\}$, then we call the set $\{x \in \mathcal{X} \mid g(x) = 0\}$ the* ***decision surface*** *or* ***decision boundary*** *of the classifier f.*

Using this definition we say that a classifier is linear if its decision surface is linear (hyperplane) and nonlinear if the decision surface has some other structure.

We will discuss three methods in this chapter: Linear Discriminant Analysis, Logistic Regression and Support Vector Machines (SVM). All of them will yield a linear classifier. However, all three methods are following different objectives in the construction of the classifier and therefore will generally yield different results.

4.2.1 Linear Discriminant Analysis

Name:	Linear Discriminant Analysis (LDA),
Type:	Supervised learning,
Input space:	$\mathcal{X} = \mathbb{R}^d$,
Output space:	$\mathcal{Y} = \{-1, 1\}$,
Function class $\mathcal{F}$:	Linear functions, $\mathcal{F} = \{\operatorname{sign}(\langle w, \phi(x)\rangle + b) \mid w \in \mathbb{R}^D,\ b \in \mathbb{R}\}$,
Loss:	squared loss $L(y, f(x)) = (y - f(x))^2$,
Regularizer:	none

We start the discussion of linear classification methods with the classic **Linear Discriminant Analysis** (LDA) or often called **Fisher Discriminant Analysis**, named after its inventor Ronald A. Fisher, the "father" of parametric statistics.

One viewpoint on linear classification is that the data enters the classifier only via the inner product $\langle w, x\rangle$ with the weight vector. In particular all x which lie on a line orthogonal to the line $L = \{\alpha w \mid \alpha \in \mathbb{R}\}$ will have the same inner product. Let $W^\top = \{z \in \mathbb{R}^D \mid \langle w, z\rangle = 0\}$, then if $x_1 - x_2 \in W^\top$, we have

$$\langle w, x_1\rangle = \langle w, x_2 + (x_1 - x_2)\rangle = \langle w, x_2\rangle + \langle w, x_1 - x_2\rangle = \langle w, x_2\rangle .$$

Basically, we have a two-step procedure. First we are projecting the feature space $\mathbb{R}^D$ onto the line L and then classify the data. Now we can ask what is the best projection $w \in \mathbb{R}^d$ in the sense that it separates the two classes in an optimal way. We define the **center of mass** or the **centroid** m_+ and m_- of the positive and negative class as:

$$m_+ = \frac{1}{n_+} \sum_{\{i \mid Y_i = 1\}} X_i, \qquad m_- = \frac{1}{n_-} \sum_{\{i \mid Y_i = -1\}} X_i,$$

where $n_+ = |\{i \mid Y_i = 1\}|$ and $n_- = |\{i \mid Y_i = -1\}|$ are the number of samples of the positive and negative class. Note that the mean of the projections of the positive and negative class is given by $\langle w, m_+\rangle$ and $\langle w, m_-\rangle$. The **within-class covariance** of the projections of the positive and negative class are given by

$$\sigma^2_{w,+} = \sum_{\{i \mid Y_i = 1\}} \Big(\langle w, X_i\rangle - \langle w, m_+\rangle \Big)^2, \qquad \sigma^2_{w,-} = \sum_{\{i \mid Y_i = -1\}} \Big(\langle w, X_i\rangle - \langle w, m_-\rangle \Big)^2.$$

The **Fisher criterion** is then defined to be the ratio of the squared difference between the two projections of the class centroids and the sum of the within-class variances of the projections:

$$J(w) = \frac{\langle w, m_+ - m_-\rangle^2}{\sigma^2_{w,+} + \sigma^2_{w,-}}.$$

Intuitively, this means that we are interested in a projection w, such that the class centroids are far away and at the same time the variances of the projections are small, as illustrated in Figure 4.5. Note, that the Fisher criterion is invariant with respect to scaling of w, we have $J(w) = J(\alpha w)$

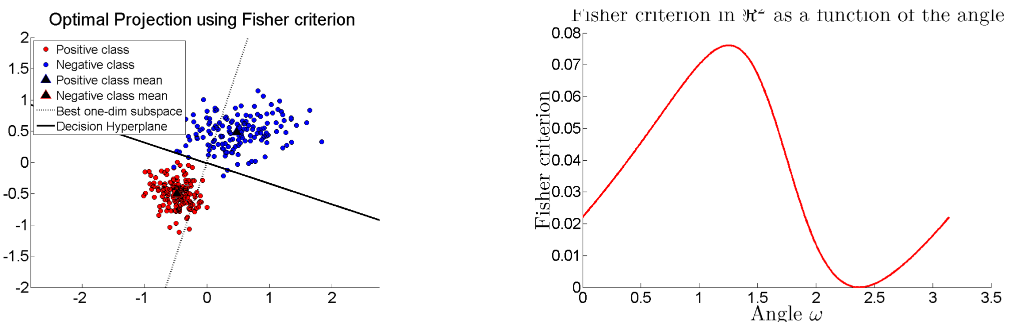


Figure 4.5: Left: A training set of two classes together with the projection w optimizing the Fisher criterion and the optimal projection line $\{\alpha w + \frac{1}{2}(m_+ + m_-) \,|\, \alpha \in \mathbb{R}\}$. Right: The Fisher criterion as a function of the angle ω, where ω is a parameterization of all weight vectors $w = (\cos(\omega), \sin(\omega))$ in $\mathbb{R}^2$.

for any $\alpha \in \mathbb{R}$, which again stresses that we are interested in finding a direction. The scaling invariance also guarantees that the criterion makes sense, because otherwise we could achieve $\lim_\alpha J(\alpha w) = \infty$, by either taking $\alpha \to 0$ or $\alpha \to \infty$. The matrix formulation of the Fisher criterion allows us to derive the optimal projection. We define the **between-class covariance** matrix C_B as

$$C_B = (m_+ - m_-)(m_+ - m_-)^T,$$

and the total **within-class covariance** matrix C_W as

$$C_W = \sum_{\{i \,|\, Y_i = 1\}} (X_i - m_+)(X_i - m_+)^T + \sum_{\{i \,|\, Y_i = -1\}} (X_i - m_-)(X_i - m_-)^T.$$

Then the Fisher criterion $J(w)$ can be written as

$$J(w) = \frac{\langle w, C_B w \rangle}{\langle w, C_W w \rangle}.$$

This criterion can be directly optimized. We emphasize again that $J(w)$ is invariant with respect to rescaling of w, that is we are only interested in the direction of w^*.

Lemma 1 *The optimal projection* $w^* = \arg\max_{w \in \mathbb{R}^D} J(w)$ *is given by*

$$w^* = C_W^{-1}(m_+ - m_-).$$

Proof: We have

$$\nabla_w J(w) = 2\frac{1}{\langle w, C_W w \rangle} C_B w - 2\frac{\langle w, C_B w \rangle}{\langle w, C_W w \rangle^2} C_W w.$$

We solve for the extrema of $J(w)$ and get

$$\frac{\langle w, C_W w \rangle}{\langle w, C_B w \rangle} C_B w = C_W w.$$

Now, $C_B w$ is always proportional to $m_+ - m_-$ and $\frac{\langle w, C_W w \rangle}{\langle w, C_B w \rangle}$ is just a scalar factor. Therefore we get that the solution is given by

$$w^* \propto C_W^{-1}(m_+ - m_-).$$

It can be checked that the Hessian of $J(w)$ at w^* is negative-definite so w^* is a maximum. $\square$

Up to now we have only found a projection which separates the data optimally in the sense of the

Fisher criterion. For the final classification we have to determine a threshold b so that our linear classifier looks like

$$f(x) = \mathrm{sign}(\langle w, x\rangle + b).$$

Usually **the threshold b is determined by minimizing the training error**.

The construction of the classifier was mainly motivated by geometrical properties of the training data, which does not fit really in our scheme of empirical and regularized empirical risk minimization. It is therefore nice to see that one can derive LDA with a particular choice of the threshold from empirical risk minimization using the squared loss.

Lemma 2 *The projection direction $w^* = C_W^{-1}(m_+ - m_-)$ obtained by maximizing the Fisher criterion is proportional to the weight vector w obtained by minimizing the least squares loss with the affine function $f(x) = \langle w, x\rangle + w_0$. Let*

$$(w', w_0') = \operatorname*{arg\,min}_{w\in\mathbb{R}^D, w_0\in\mathbb{R}} \sum_{i=1}^{n} (Y_i - \langle w, X_i\rangle - w_0)^2.$$

Then $w' \sim w^*$.

Proof: Exercise ! □

The (special) least squares solution also determines the threshold w_0 for the classifier. Generally, the manual choice of the threshold b by optimization of the training error yields better results since it directly minimizes the error count instead of the least squares loss which is a quite bad approximation of the 0-1-loss (see also later discussion when we compare LDA to logistic regression).

In the remainder we want to discuss the potential of LDA as a **dimensionality reduction** method if one has more than two classes. As indicated above the LDA can be understood as summarizing the data in a one-dimensional subspace such that the two classes are optimally separated. Therefore LDA can also be seen as a dimensionality reduction method. Now, if one has more than two classes than a projection onto a one-dimensional subspace will generally not do a good job. For a general dataset of K classes which are linearly separable in $\mathbb{R}^d$ we need $K-1$ projections. The generalization of LDA to the multi-class case is quite easy, see also Duda et al. (2000). Let $m = \frac{1}{n}\sum_{i=1}^n X_i$ be the mean of all data points. Then the general within-class covariance matrix S_W is just the sum of the covariance matrices of each class (but note that they are not normalized !)

$$C_W = \sum_{k=1}^{K} \sum_{\{i\,|\,Y_i=k\}} (X_i - m_k)(X_i - m_k)^T,$$

where m_k is the centroid of the k-th class, $m_k = \frac{1}{n_k}\sum_{\{i\,|\,Y_i=k\}} X_i$, and $n_k = |\{i\,|\,Y_i = k\}|$ is the number of samples of class k. The generalization of the between-class covariance matrix C_B for K classes is not so straightforward. Basically the motivation is to decompose the covariance C of the whole data into two parts: the between-class covariance and the within-class covariance. The definition of our generalized within-class covariance C_W was very natural. The definition of the between-class covariance follows now directly from the following decomposition of the covariance matrix.

$$\begin{aligned}
C = \sum_i (X_i - m)(X_i - m)^T &= \sum_{k=1}^{K} \sum_{\{i\,|\,Y_i=k\}} (X_i - m_k + m_k - m)(X_i - m_k + m_k - m)^T \\
&= \sum_{k=1}^{K} \sum_{\{i\,|\,Y_i=k\}} (X_i - m_k)(X_i - m_k)^T + \sum_{k=1}^{K} n_k (m_k - m)(m_k - m)^T. \\
&= C_B + C_W,
\end{aligned}$$

where we have identified the first term as between-class covariance matrix C_B.

$$C_B = \sum_{k=1}^{K} n_k (m_k - m)(m_k - m)^T.$$

As stated we would like to find a $K-1$-dimensional subspace of $\mathbb{R}^d$ of the original input space in which we project the data. Our original way to derive the optimal projection leads to a complicated optimization problem. However, by a closer inspection of the Fisher criterion one observes that one can also derive directly a multi-dimensional projection. The first key observation is that C_B and C_W are symmetric matrices and therefore maximizing

$$J(w) = \frac{\langle w, C_B w\rangle}{\langle w, C_W w\rangle}.$$

corresponds to the generalized Rayleigh-Ritz principle which is a variational formulation of the **generalized eigenvalue problem**, see Section B.2 for the definition. One can deduce that the optimal $K-1$-dimensional subspace corresponds to the subspace spanned by the $K-1$ eigenvectors of the generalized eigenvalue problem $C_B w = \lambda\, C_W w$ corresponding to the $K-1$ largest eigenvalues [4]. Note that if C_W is invertible, it corresponds to the eigenvalue problem $C_W^{-1} C_B w = \lambda w$. The advantage of LDA as a dimensionality reduction method over the related unsupervised PCA, see Section 10.1, is that it takes into account the label information in the data.
Remark:

- In Proposition 1 we derived that a transformation of the data never can improve the Bayes risk of the underlying classification problem. It is important to note that a linear projection of the data will in generally lead to a worse Bayes risk (in particular if the data is **not** linearly separable). Why are we doing then dimensionality reduction at all ? In high dimension the problem might be very difficult to solve (curse of dimensionality) and the hope is that by doing dimensionality reduction we can at least find a relatively good solution in this low-dimensional subspace.

- One should always be cautious when the number of training points for LDA is on the same order as the number of features. In this case it can easily happen that LDA overfits the data.

4.2.2 Logistic regression

Name:	Logistic Regression,
Type:	Supervised binary classification,
Input space:	$\mathcal{X} = \mathbb{R}^d$,
Output space:	$\mathcal{Y} = \{-1, 1\}$,
Function class $\mathcal{F}$:	$\mathcal{F} = \{\mathrm{sign}(\langle w, \phi(x)\rangle + b) \,\vert\, w \in \mathbb{R}^D,\, b \in \mathbb{R}\}$,
Loss function:	logistic loss $L(y, f(x)) = \log_2(1 + \exp(-y\,(\langle w, \phi(x)\rangle + b)))$,
Regularizer:	standard none but regularized version is discussed

Despite its name **logistic regression** is actually a learning method for binary classification.

Definition 15 *Given a training sample $T_n = (X_i, Y_i)_{i=1}^n$ with $X_i \in \mathcal{X}$ and $Y_i \in \{-1, 1\}$ and the function space $\mathcal{F} = \{\sum_{i=1}^D w_i \phi_i(x) \,\vert\, w \in \mathbb{R}^D\}$ we define **logistic regression** as the mapping $\mathcal{A} : T_n \to \mathcal{F}$ with,*

$$T_n \mapsto f_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \log\Big(1 + \exp(-Y_i\, \langle w, \phi(X_i)\rangle)\Big). \tag{4.8}$$

Thus logistic regression is just doing empirical risk minimization using the logistic loss and a class of functions which is linear in its parameter w. In its standard formulation logistic regression is the maximum likelihood solution using the following model for the conditional probability:

$$\mathrm{P}(Y = 1 | X = x,\, w) = \frac{1}{1 + e^{-\langle w, \phi(x)\rangle}},$$

[4] All other eigenvalues will be zero. Note, that $m = \frac{1}{n}\sum_{k=0}^K n_k m_k$ and thus $\sum_{k=0}^K n_k(m_k - m) = \sum_{k=0}^K n_k m_k - nm = 0$. Therefore the set of vectors $m_i - m, i = 1, \ldots, K$ is linearly dependent and C_B has rank $K-1$ given that $m_1, \ldots, m_K$ are linearly independent.

as we have discussed in Section 2.3.1. The model assumption is that the log ratio of the conditional probabilities is a linear function

$$\log\Big(\frac{\mathrm{P}(Y=1|X=x,\,w)}{\mathrm{P}(Y=-1|X=x,\,w)}\Big) = \langle w, \phi(x)\rangle + b.$$

As noted in Chapter 3 the maximum likelihood solution is equivalent to the solution of empirical risk minimization given that we have $L(y, f(x)) = -\log p(y|x, f)$. Note as discussed in least squares regression we integrate the constant b by extending the input space. For simplicity we therefore discard the constant offset in the following.

Unfortunately, the solution of logistic regression cannot be found analytically. We have to resort to an iterative optimization technique. Naturally, we could take any of our favorite optimization techniques to solve the problem in Equation (4.8). However, since there is a common standard technique to solve logistic regression we discuss this method, the so called **iterative reweighted least squares**, in detail.

First we derive the gradient and the Hessian of the empirical risk

$$R_{\mathrm{emp}}(w) = \frac{1}{n}\sum_{i=1}^{n} \log\Big(1+\exp(-Y_i\,\langle w, \phi(X_i)\rangle)\Big),$$

as

$$\frac{\partial R_{\mathrm{emp}}}{\partial w_s}(w) = -\frac{1}{n}\sum_{i=1}^{n} y_i\,\phi_s(X_i)\frac{\exp(-Y_i\,\langle w,\phi(X_i)\rangle)}{1+\exp(-Y_i\,\langle w,\phi(X_i)\rangle)},$$

$$\frac{\partial^2 R_{\mathrm{emp}}}{\partial w_r \partial w_s}(w) = \frac{1}{n}\sum_{i=1}^{n} \phi_s(X_i)\phi_r(X_i)\frac{\exp(-Y_i\,\langle w,\phi(X_i)\rangle)}{\Big(1+\exp(-Y_i\,\langle w,\phi(X_i)\rangle)\Big)^2}.$$

We will derive the solution using a gradient descent method, which is an iterative method to find the *local* extrema of a function, see Section D in the appendix for an introduction to solving unconstrained optimization problems. A specific gradient decent method is the so called **Newton-Raphson** algorithm, which has the following update rule

$$w_{\mathrm{new}} = w_{\mathrm{old}} - \Big(\frac{\partial^2 R_{\mathrm{emp}}}{\partial w_r \partial w_s}(w)\Big)^{-1} \nabla_w R_{\mathrm{emp}}(w).$$

This is basically a Newton descent step with the stepsize fixed to 1. Note that with the diagonal matrices W and D with diagonal entries

$$W_{ii} = \frac{\exp(-Y_i\,\langle w,\phi(X_i)\rangle)}{(1+\exp(-Y_i\,\langle w,\phi(X_i)\rangle))^2}, \qquad D_{ii} = \frac{\exp(-Y_i\,\langle w,\phi(X_i)\rangle)}{1+\exp(-Y_i\,\langle w,\phi(X_i)\rangle)},$$

we can write the gradient and Hessian $H(R_{\mathrm{emp}})\big|_w$ of R_{emp} at w as

$$\nabla_w R_{\mathrm{emp}}(w) = -\frac{1}{n}\Phi^T D Y, \qquad H(R_{\mathrm{emp}})\big|_w = \frac{1}{n}\Phi^T\,W\,\Phi.$$

Thus we can write the Newton-Raphson update as

$$\begin{aligned} w_{\mathrm{new}} &= w_{\mathrm{old}} + \Big(\Phi^T\,W\,\Phi\Big)^{-1}\Phi^T D Y = \Big(\Phi^T\,W\,\Phi\Big)^{-1}\Phi^T W\Big(\Phi w_{\mathrm{old}} + W^{-1} D Y\Big) \\ &= \Big(\Phi^T\,W\,\Phi\Big)^{-1}\Phi^T W Z, \end{aligned}$$

where the factors $\frac{1}{n}$ cancel since $\Big(\frac{1}{n}\Phi^T\,W\,\Phi\Big)^{-1} = n\Big(\Phi^T W \Phi\Big)^{-1}$ and we have introduced $Z = \Phi w_{\mathrm{old}} + W^{-1} D Y$. Note that the equation in the last line is the solution of a *weighted* least squares problem with desired outputs Z. The objective function of the weighted least squares problem using weights, γ_i, $i = 1, \ldots, n$, with $\gamma_i > 0$ is given by

$$\sum_{i=1}^{n} \gamma_i (Y_i - \langle w, \Phi(X_i)\rangle)^2 = \langle Y - \Phi w, \Gamma(Y - \Phi w)\rangle\,,$$

where Γ is the diagonal weight matrix with weights γ_i on the diagonal and we have omitted the normalization $\frac{1}{n}$. The derivation of the solution can be derived from ridge regression and yields the weight vector

$$w = \Big(\Phi^T \, \Gamma \, \Phi\Big)^{-1} \Phi^T \Gamma Y.$$

Comparing this expression with the Newton-Raphson update we get $\Gamma = W$ and $Y = Z$. Since each step in the Newton-Raphson method corresponds to solving a weighted least squares problem and the weights change in each iteration, this method to solve the optimization problem of logistic regression is called **iteratively reweighted least squares**.

The problem with logistic regression is that it is a method only based on empirical risk minimization and therefore prone to overfitting. Moreover, for a linearly separable dataset the solution w is unbounded. As we have done already in ridge regression we can resolve this problem by adding a regularizer on the weights, in this case we use for computational convenience the squared regularizer. This makes also the numerical solution more stable since the involved matrices might be close to singular.

Definition 16 *Given a training sample $T_n = (X_i, Y_i)_{i=1}^n$ with $X_i \in \mathcal{X}$ and $Y_i \in \{-1, 1\}$ and the function space $\mathcal{F} = \{\sum_{i=1}^D w_i \phi_i(x) \,|\, w \in \mathbb{R}^D\}$ we define L_2-**regularized logistic regression** as the mapping $\mathcal{A} : T_n \to \mathcal{F}$ with,*

$$T_n \mapsto f_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \log\Big(1 + \exp(-Y_i \, \langle w, \phi(X_i)\rangle)\Big) + \lambda \, \|w\|_2^2 \,, \tag{4.9}$$

where λ is the regularization parameter.

The gradient and the Hessian of the objective function

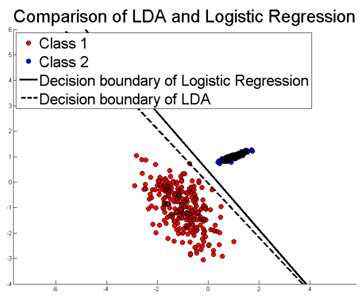
$$R_\lambda(w) = \frac{1}{n} \sum_{i=1}^n \log\Big(1 + \exp(-Y_i \, \langle w, \phi(X_i)\rangle)\Big) + \lambda \, \|w\|_2^2 \,,$$

change in the following way

$$\begin{aligned}
\frac{\partial R_\lambda}{\partial w_s}(w) &= -\frac{1}{n} \sum_{i=1}^n Y_i \, \phi_s(X_i) \frac{\exp(-Y_i \, \langle w, \phi(X_i)\rangle)}{1 + \exp(-Y_i \, \langle w, \phi(X_i)\rangle)} + 2\lambda w_s, \\
\frac{\partial^2 R_\lambda}{\partial w_r \partial w_s}(w) &= \frac{1}{n} \sum_{i=1}^n \phi_s(X_i)\phi_r(X_i) \frac{\exp(-Y_i \, \langle w, \phi(X_i)\rangle)}{\Big(1 + \exp(-Y_i \, \langle w, \phi(X_i)\rangle)\Big)^2} + 2\lambda \mathbb{1}_D,
\end{aligned}$$

where $\mathbb{1}_D$ is the $D \times D$-identity matrix. One can easily solve this problem by plugging the above expressions of the gradient and Hessian in the Newton-Raphson scheme.

In Figure 4.6 we compare the results of LDA and Logistic regression for two datasets (for numerical stability we use the regularized version of logistic regression with a small regularization parameter $\lambda = 0.01$). In both cases the class conditional distributions are Gaussians. Often the results for this data look quite similar. The main difference between LDA and Logistic regression is that in LDA the global data structure has a larger influence than for Logistic Regression. This can be clearly seen from the second Figure 4.7 where we have added additional data for the positive class which does *not* change the optimal decision boundary. Nevertheless, since LDA takes into account *all* the data the decision boundary of LDA from the original to the perturbed data changes completely and actually the chosen classifier for the perturbed data is pretty bad. This is in contrast to logistic regression which is very stable against the perturbation. The reason for this is that the least squares loss is influenced heavily by training data which lies far away from the decision boundary, whereas the logistic loss quickly decays far away from the decision boundary and is therefore only marginally influenced by new training data far away from the decision boundary.



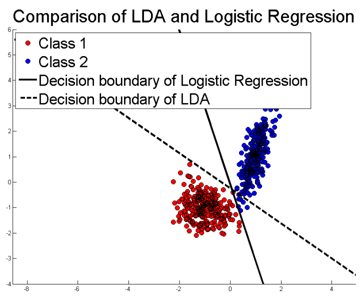
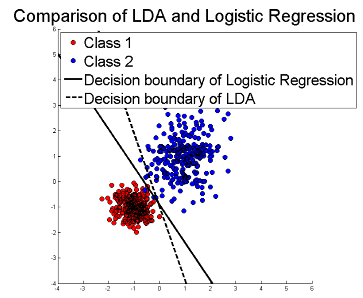


Figure 4.6: Left: A linearly separable problem, Right: A non-separable problem.



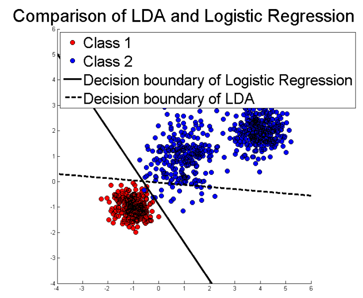


Figure 4.7: Left: Original data, Right: Adding the second Gaussian blob should not change the decision boundary. However, LDA changes its decision completely.

4.2.3 Support Vector Machines

The linear **support vector machine** can be motivated from different perspectives. We will start with the geometric point of view. Up to now we have discussed two ways of determining a linear classifier. Both LDA and Logistic Regression find an optimal hyerplane which separates the data with respect to some criterion, in the case of LDA it is the Fisher score, for logistic regression it is the logistic loss. We will now define the **maxium margin hyperplane** for linearly separable data. This hyerplane is optimal in the sense that it is the unique hyerperplane which correctly classifies the data and has maximal distance from the training data. This is the so-called **hard margin** case before. Later on we will relax the condition that the data has to be linearly separable which leads to the so-called **soft margin** case. However, for the later case it is harder to provide a geometric interpretation.

A linear hyperplane is fully determined by the weight vector w and the offset b. However, finally we are only interested in the sign of the hyperplane, or said otherwise we are only interested in the classifier defined as,

$$f(x) = \text{sign}(\langle w, x\rangle + b).$$

In particular the decision boundary, $\langle w, x\rangle + b = 0$, is the most interesting quantity, since it tells us where we change our decision from one class to the other class. It is obvious that there exist many values of the parameters w, b which lead to the same classifier. Namely, multiplication of the weights and the offset with a positive factor γ, $\tilde{w} = \gamma w$ and $\tilde{b} = \gamma b$ will yield a new weight vector and offset which yields an equivalent classifier. We will now fix this degree of freedom by defining the **canonical hyperplane**.

Definition 17 *The pair $(w, b) \in \mathbb{R}^d \times \mathbb{R}$ is said to be in canonical form with respect to $x_1, \ldots, x_n \in \mathbb{R}^d$, if it scaled such that*

$$\min_{i=1,\ldots,n} |\langle w, x_i\rangle + b| = 1,$$

Name:	Linear Support Vector Machine (SVM),
Type:	Supervised learning,
Input space:	$\mathcal{X} = \mathbb{R}^d$,
Output space:	$\mathcal{Y} = \{-1, 1\}$,
Function class $\mathcal{F}$:	Linear functions, $\mathcal{F} = \{\text{sign}(\langle w, x\rangle + b) \,\vert\, w \in \mathbb{R}^d \,, b \in \mathbb{R}\}$,
Loss:	hinge loss $L(y, f(x)) = \max\{0, 1 - y\, f(x)\}$,
Regularizer:	L_2-regularization $\Omega(f) = \sum_{i=1}^d w_i^2 = \Vert w\Vert_2^2$.

*which implies that the point closest to the hyperplane has distance $\frac{1}{\|w\|}$. We call $\rho = \frac{1}{\|w\|}$ the **geometrical margin** of the hyperplane h=$\{x \mid \langle w, x\rangle + b = 0\}$.*

In Figure 4.8 a hyperplane in canonical form is shown. The **maximum margin hyperplane**, a

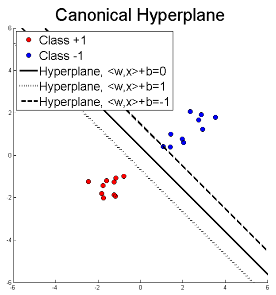


Figure 4.8: The canonical hyperplane for a set of training points $(X_i)_{i=1}^n$.

hyperplane which correctly classifies the data and has maximum distance/margin to the data, is now defined as follows

Definition 18 *A maximum margin hyperplane (w, b) for a linearly separable set of training data $(X_i, Y_i)_{i=1}^n$ is defined as*

$$\max_{w\in\mathbb{R}^d,\, b\in\mathbb{R}} \min\{\|x - X_i\| \mid \langle w, x\rangle + b = 0,\, x \in \mathbb{R}^d,\ i = 1, \ldots, n\},$$

where we optimize over all (w, b) such that $Y_i\,(\langle w, X_i\rangle + b) > 0$.

Note that $\frac{|\langle w,y\rangle + b|}{\|w\|}$ is the distance of a point y to a hyperplane $\{x \mid \langle w, x\rangle + b = 0\}$. For the derivation of this result as an example for the tools in Section D in the appendix see (10.11). Note that as the classifier, $f(x) = \text{sign}(\langle w, x\rangle + b)$, the maximum margin hyperplane is also not uniquely defined since a scaling as above will yield the same hyperplane. We can fix this degree of freedom by using the canonical hyperplane. Noting that $\frac{1}{\|w\|}$ is the distance of the canonical hyperplane to the closest data point, the problem of finding the maximum margin hyperplane can be formulated as

$$\max_{w\in\mathbb{R}^d,\, b\in\mathbb{R}} \frac{1}{\|w\|}$$
$$\text{subject to: } Y_i(\langle w, X_i\rangle + b) \geq 1, \quad \forall\, i = 1, \ldots, n$$

Typically, the problem is transferred into the equivalent problem:

$$\min_{w\in\mathbb{R}^d,\, b\in\mathbb{R}} \frac{1}{2}\|w\|^2 \tag{4.10}$$
$$\text{subject to: } Y_i(\langle w, X_i\rangle + b) \geq 1, \quad \forall\, i = 1, \ldots, n$$

In order to derive further properties of this problem which will also clarify the notion of **support vectors** we need several facts from optimization theory, summarized in Section D of the appendix.

First of all we note that the optimization problem (4.10) is convex since the objective function is quadratic in w and the inequality constraints are linear in w and b. More specifically it is a quadratic programming problem. The associated Lagrangian is given by

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^n \alpha_i \Big[1 - Y_i(\langle w, X_i\rangle + b)\Big],$$

where $\alpha_i \geq 0, \quad \forall\, i = 1,\ldots,n$. Remember, that the dual Lagrangian $q(\alpha)$ is given by

$$q(\alpha) = \inf_{w\in\mathbb{R}^d,\, b\in\mathbb{R}} L(w,b,\alpha).$$

Since $L(w,b,\alpha)$ is convex we can derive the global optimum for fixed α by computing the stationary point

$$\nabla_w L(w,b,\alpha) = w - \sum_{i=1}^n \alpha_i Y_i X_i, \qquad \frac{\partial L(w,b,\alpha)}{\partial b} = -\sum_{i=1}^n \alpha_i Y_i.$$

Thus we get the two conditions

$$w = \sum_{i=1}^n \alpha_i Y_i X_i, \qquad \sum_{i=1}^n \alpha_i Y_i = 0.$$

Note, that the weight vector w is expressed in terms of the data points. When we use SVMs with kernels this relation will become crucial. Plugging these expressions into $L(w,b,\alpha)$ we get the dual Lagrangian

$$\begin{aligned} q(\alpha) &= \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j Y_i Y_j \langle X_i, X_j\rangle + \sum_{i=1}^n \alpha_i - \sum_{i,j=1}^n \alpha_i\alpha_j Y_i Y_j \langle X_i, X_j\rangle \\ &= -\frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j Y_i Y_j \langle X_i, X_j\rangle + \sum_{i=1}^n \alpha_i, \end{aligned}$$

where $\alpha_i \geq 0, \quad \forall\, i = 1,\ldots,n$. Hence, the dual problem is given by

$$\begin{aligned} &\max_{\alpha\in\mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j Y_i Y_j \langle X_i, X_j\rangle, \qquad (4.11)\\ &\text{subject to: } \alpha_i \geq 0, \quad i = 1,\ldots,n, \\ &\sum_{i=1}^n Y_i\alpha_i = 0. \end{aligned}$$

The last constraint $\sum_{i=1}^n Y_i\alpha_i = 0$ is a feasibility constraint since if $\sum_{i=1}^n Y_i\alpha_i \neq 0$ we can achieve $q(\alpha) \to -\infty$ by letting b tend to plus or minus infinity. It can be easily seen that Slater's condition is valid for this convex optimization problem given that one has at least one training point from each class. That implies that strong duality holds: the optimal value of the primal and dual problem are equal. Therefore solving the dual problem (4.11) is equivalent to solving the primal problem (4.10). Usually one solves in practice the dual problem. The most often used method is the SMO (Sequential Minimal Optimization) algorithm proposed by Platt. The complexity depends on the dataset, the easier to separate the faster it can be solved. Alternatively, you can use any solver for quadratic programs in particular the interior point method discussed in D.3. In the worst case solving the optimization problem (4.11) has complexity $O(n^3)$.

One can show that the dual problem (4.11) is equivalent to finding the distance between the convex hulls of positive and negative class:

Lemma 3 *The optimization problem* (4.11) *is equivalent to finding the distance of the convex hulls of positive and negative class:*

$$\min_{\alpha\in\mathbb{R}^n} \left\| \sum_{i=1,Y_i=1}^{n} \alpha_i X_i - \sum_{j=1,Y_j=-1}^{n} \alpha_j X_j \right\|^2, \tag{4.12}$$

$$\textit{subject to: } \alpha_i \geq 0, \quad i=1,\ldots,n,$$

$$\sum_{i=1,Y_i=1}^{n} \alpha_i = \sum_{j=1,Y_j=-1}^{n} \alpha_j = 1.$$

Proof: For the optimization problem (4.11) scaling α_i with a positive constant does not change the constraints. We use that to eliminate this scaling degree of freedom. We introduce new variables β_i and γ_i as

$$\beta_i = \frac{2\alpha_i}{\sum_{j=1}^n \alpha_j} \qquad \text{and } \gamma = \sum_{j=1}^n \alpha_j.$$

Thus the dual problem can be reformulated as

$$\max_{\beta\in\mathbb{R}^n,\gamma\in\mathbb{R}} \quad \gamma - \frac{\gamma^2}{8} \left\| \sum_{i=1,Y_i=1}^{n} \beta_i X_i - \sum_{j=1,Y_j=-1}^{n} \beta_j X_j \right\|^2,$$

$$\text{subject to: } \beta_i \geq 0, \quad i=1,\ldots,n, \qquad \sum_{i=1}^n Y_i\beta_i = 0, \qquad \sum_{i=1}^n \beta_i = 2$$

Note, that γ is unconstrained, so we can maximize the objective directly with respect to γ. The maximum of

$$\gamma - \frac{\gamma^2}{8}c,$$

where $c = \left\| \sum_{i=1,Y_i=1}^n \beta_i X_i - \sum_{j=1,Y_j=-1}^n \beta_j X_j \right\|^2$ is attained at $\gamma = \frac{4}{c}$. The new objective becomes $\frac{2}{c}$. We use that to turn the problem into a minimization problem and just minimize c. This leads to the problem given in 4.12. □

The geometric problem is illustrated in Figure 4.9. We finish the characterization of the SVM by stating the Karush-Kuhn-Tucker conditions for the quadratic program (4.10). The most important one is the complementary slackness condition:

$$\alpha_i \Big[1 - Y_i(\langle w, X_i\rangle + b) \Big] = 0,$$

which yields

$$\alpha_i > 0 \quad \text{if} \quad \Big[1 - Y_i(\langle w, X_i\rangle + b)\Big] = 0 \qquad \text{and} \qquad \Big[1 - Y_i(\langle w, X_i\rangle + b)\Big] < 0 \quad \text{if} \quad \alpha_i = 0.$$

In particular, only points which lie on the margin (the training points closest to the decision boundary), $\Big[1 - Y_i(\langle w, X_i\rangle + b)\Big] = 0$, have non-zero weights and are called **support vectors** since they support the hyperplane.

The offset b can thus be determined by averaging the value $b = Y_i - \langle w, X_i\rangle$ over all points with $\alpha_i > 0$:

$$b = \frac{1}{\sum_{i=1}^n \mathbb{1}_{\alpha_i>0}} \sum_{i=1}^n \mathbb{1}_{\alpha_i>0}(Y_i - \sum_{j=1}^n \alpha_j\, Y_j\, \langle X_i, X_j\rangle),$$

where we have used the derived form of the weight vector w, $w = \sum_{i=1}^n \alpha_i Y_i X_i$. Therefore only the points in the training data with $\Big[1 - Y_i(\langle w, X_i\rangle + b)\Big] = 0$, which are the points closest to the decision boundary, will contribute to the weight vector w. The area between the two supporting hyperplanes $\{x \mid \langle w, x\rangle + b = 1\}$ and $\{x \mid \langle w, x\rangle + b = -1\}$ is called the **margin**. The complementary slackness conditions have two important implications:

1. The weight vector of the support vector machine is usually **sparse**, in the sense that only a few components of α are non-zero and thus only a few training points contribute to the decision boundary. This is in contrast to logistic regression where usually all training points contribute with a non-zero coefficient to the weight vector.

2. The Support Vector Machine (SVM) is quite robust to perturbations of the data, since a modification of a training point does not change the weight vector and thus the classifier as long as the modified point does not move inside the margin.

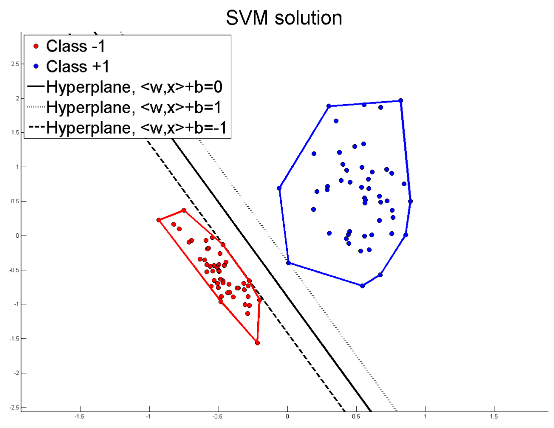


Figure 4.9: A linearly separable problem. The hard margin solution of the SVM is shown together with the convex hulls of the positive and negative class. The points on the margin, that is $\langle w, x\rangle + b = \pm 1$, are called **support vectors**.

Obviously not all data is linearly separable. Then the maximum margin hyperplane does not exist. Moreover, the **hard margin** case is often too strict since we do not allow any errors of our classifier which can lead to overfitting of the training data. This has lead to the development of a relaxed version of the maximum margin principle which can also be cast into our framework of regularized empirical risk minimization. This is the so called **soft margin** case. This case is usually meant if one speaks about the "support vector machine". The hard margin case is in practice almost never used. However, the reason for presenting the hard margin case is that it has an intuitive geometric interpretation which one can use as a guideline for the soft margin case.

The soft margin case has this name because we do not enforce anymore the constraint $Y_i(\langle w, X_i\rangle + b) \geq 1$ but relax it using so called **slack variables** $\xi_i \geq 0$. However, we penalize this relaxation by adding a term for the slack variables to the objective function

$$\min_{w\in\mathbb{R}^d,\, b\in\mathbb{R},\, \xi\in\mathbb{R}^n} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^n \xi_i \tag{4.13}$$

$$\text{subject to: } Y_i(\langle w, X_i\rangle + b) \geq 1 - \xi_i, \quad \forall\, i = 1, \ldots, n,$$
$$\xi_i \geq 0, \quad \forall\, i = 1, \ldots, n \tag{4.14}$$

where the constant $C > 0$ is often called error parameter. Thus we have now a tradeoff between letting points into the margin or even misclassifying training points and a large margin.

Now, the above optimization problem can also be cast into our framework of regularized empirical risk minimization. Namely note that we can rewrite the constraint using that $\xi_i \geq 0$,

$$\xi_i \geq \max\Big(0, 1 - Y_i(\langle w, X_i\rangle + b)\Big).$$

Since we penalize the slack variables ξ_i, we will have $\xi_i = \max\Big(0, 1 - Y_i(\langle w, X_i\rangle + b)\Big)$ at the optimal point[5]. Now with $f(X_i) = \langle w, X_i \rangle + b$ we note that $\max\Big(0, 1 - y_i\, f(X_i)\Big)$ is nothing else than the **hinge loss** which we have encountered in Chapter 2 on Bayesian decision theory. Thus we could equivalently write the optimization problem as

$$\min_{w \in \mathbb{R}^d,\, b \in \mathbb{R}} C\, \frac{1}{n} \sum_{i=1}^{n} \max\Big(0, 1 - Y_i(\langle w, X_i \rangle + b)\Big) + \|w\|^2 ,$$

which is just regularized empirical risk minimization using the hinge loss and a L_2-regularization. Note that the parameter C is inverse to our usual regularization parameter by multiplying the whole objective function with $\frac{1}{C}$ we would get $\lambda = \frac{1}{C}$. This notation has become standard for support vector machines.

Similar to the previous case we can write down the Lagrangian

$$L(w, b, \xi, \alpha, \beta) = \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i + \sum_{i=1}^{n} \alpha_i \Big[1 - \xi_i - Y_i(\langle w, X_i \rangle + b)\Big] - \sum_{i=1}^{n} \beta_i \xi_i,$$

where $\alpha_i \geq 0,\ i = 1, \ldots, n$ and $\beta_i \geq 0,\ i = 1, \ldots, n$ and derive the dual problem of the optimization problem 4.13 by deriving the stationary point:

$$\begin{aligned}
\nabla_w L(w, b, \xi, \alpha, \beta) &= w - \sum_{i=1}^{n} \alpha_i Y_i X_i, \\
\frac{\partial L(w, b, \xi, \alpha, \beta)}{\partial b} &= -\sum_{i=1}^{n} \alpha_i Y_i, \\
\nabla_\xi L(w, b, \xi, \alpha, \beta) &= \frac{C}{n}1 - \alpha - \beta,
\end{aligned}$$

where 1 is a n-dimensional vector of ones. We get the conditions:

$$w = \sum_{i=1}^{n} \alpha_i Y_i X_i, \qquad \sum_{i=1}^{n} \alpha_i Y_i = 0, \qquad \beta = \frac{C}{n}1 - \alpha.$$

The last equation can be used to get rid of β. Since $\beta_i \geq 0$ for all $i = 1, \ldots, n$ we get an upper bound for α_i,

$$0 \leq \alpha_i \leq \frac{C}{n}, \quad i = 1, \ldots, n.$$

and thus the dual problem can be formulated as

$$\begin{aligned}
\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j Y_i Y_j \langle X_i, X_j \rangle , \qquad (4.15) \\
\text{subject to: } 0 \leq \alpha_i \leq \frac{C}{n}, \quad i = 1, \ldots, n, \\
\sum_{i=1}^{n} Y_i \alpha_i = 0.
\end{aligned}$$

The KKT conditions (complementary slackness condition) of the original problem are given by

$$\alpha_i \Big[1 - \xi_i - Y_i(\langle w, X_i \rangle + b)\Big] = 0, \qquad \text{and} \qquad \beta_j \xi_j = 0, \qquad i, j = 1, \ldots, n,$$

where the last equality is equivalent to, $\Big(\frac{C}{n} - \alpha_j\Big)\xi_j = 0$. Using these conditions we have three classes of points

[5]If the constraints are fulfilled with an inequality we can always decrease the objective function by decreasing the ξ_i so that the constraint inequalities become tight.

- $\alpha_i = 0$: These are points where $\beta_i = \frac{C}{n}$ and therefore $\xi_i = 0$. Moreover by the other KKT condition $Y_i(\langle w, X_i\rangle + b) > 1$ and thus these points lie outside the margin and are all correctly classified.
- $0 < \alpha_i < \frac{C}{n}$: These are points where $\beta_i > 0$ and therefore also $\xi_i = 0$. However, by the second KKT condition $Y_i(\langle w, X_i\rangle + b) = 1$ and thus these point lie exactly on the margin. They are all correctly classified.
- $\alpha_i = \frac{C}{n}$: For these points $\beta_i = 0$ and thus $\xi_i > 0$. If $\xi_i \leq 1$ then the points are still correctly classified, whereas for $\xi_i > 1$ the points are misclassified. Since $\alpha_i > 0$ we have $Y_i(\langle w, X_i\rangle + b) = 1 - \xi_i$.

From this analysis follows that we can compute the constant b by averaging over all points with $0 < \alpha_i < \frac{C}{n}$. The Figure 4.10 shows a soft-margin SVM for four different values of the error parameter C.

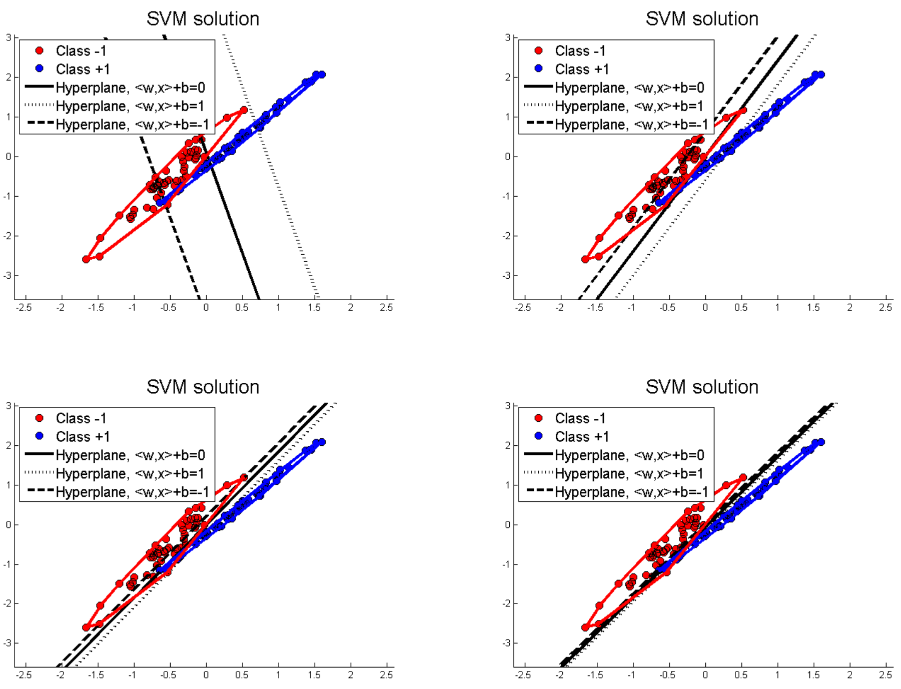


Figure 4.10: A non linearly separable problem. The soft margin solution of the SVM is shown together with the convex hulls of the positive and negative class. Top row: error parameter $C = 10^1$ (left) and $C = 10^2$ (right), Bottom row: error parameter $C = 10^3$ (left) and $C = 10^4$ (right). One observes here the trade-off between having a large margin (for small error parameter) and small loss (for large error parameter).